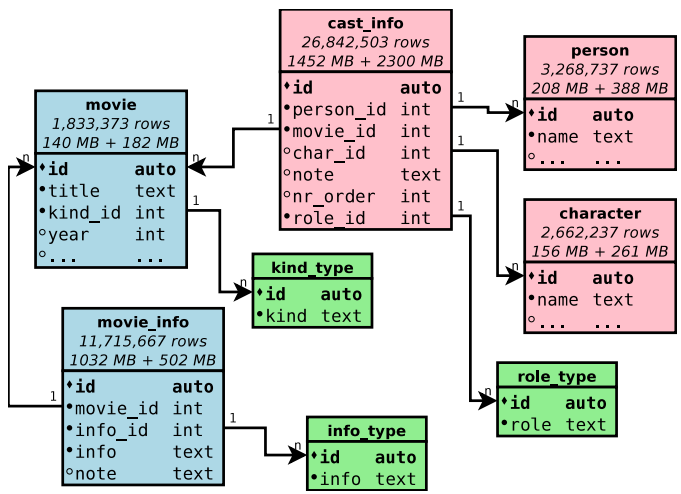
Relational Model, Key Constraints

PDBM 6.1

Dr. Chris Mayfield

Department of Computer Science
James Madison University

Feb 01, 2022



What is a data model?

Notation for describing data or information

- Structure of the data
  - Conceptual vs physical
- Operations on the data
  - Limited set of queries / updates
- Constraints on the data
  - Limitations of what data can be

Logical data models

- Relational / object-relational
- Semistructured (e.g., XML)

Relational model

title	*year*	*length*	*genre*
Gone With the Wind	1939	231	drama
Star Wars	1977	124	sci-fi
Wayne's World	1992	95	comedy

- Structure: Table (like an ArrayList of objects)
  - Columns define role played by different pieces of data
- Operations: Relational Algebra (select, project, join)
- Constraints:
  - "Genre must be action, comedy, drama, . . ."
  - "No two movies can have same title and year"

Semistructured model (XML)

Extensible Markup Language

```
<movies>
    <movie title="Gone With the Wind">
        <year>1939</year>
        <length>231</length>
        <genre>drama</genre>
    </movie>
    ...
</movies>
```

- Structure: Tree (or graph)
  - Tags define role played by different pieces of data
- Operations: Traversals in the implied tree
- Constraints:
  - Limitations on data types (per tag)
  - Which tags can be top-level / nested

Semistructured model (JSON)

JavaScript Object Notation

```
{"movies": [
    "movie": {
        "title": "Gone With the Wind",
        "year": 1939,
        "length": 231,
        "genre": "drama"
    }
    ...
]}
```

- Arrays in []'s
  - values (string, number, object, array, `true`, `false`, `null`)
- Objects in {}'s
  - key:value pairs (keys must be strings)

"The Fat-Free Alternative to XML" http://www.json.org/xml.html

The Relational Model

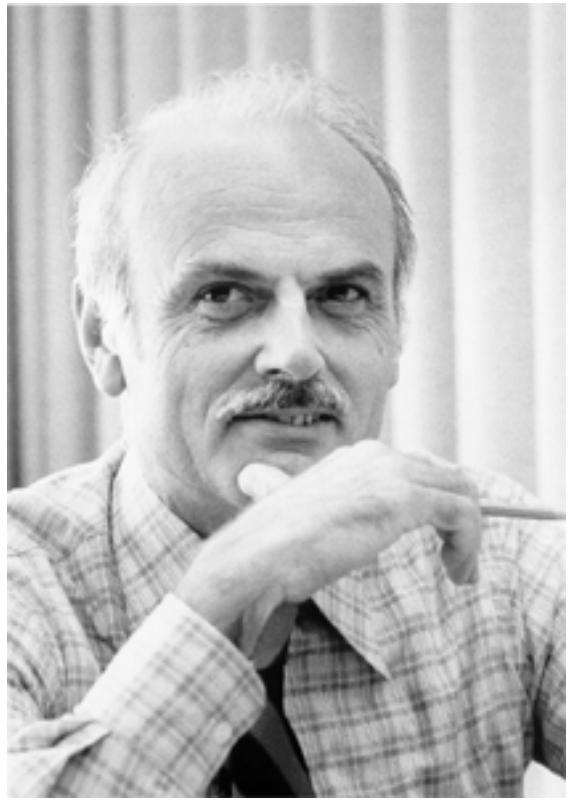
Serving databases since 1969

Terminology

title	*year*	*length*	*genre*
Gone With the Wind	1939	231	drama
Star Wars	1977	124	sci-fi
Wayne's World	1992	`NULL`	comedy

- Schema: `Movie(title, year, length, genre)`
  - *Database schema* = set of schemas of relations
  - *DBMS schema* = collection of tables, views, etc
- Relation (table), Attribute (column), Tuple (row)
- `NULL` is a special value that can mean:
  - Unknown / undefined / empty / ...

Who invented all this?



Edgar Frank "Ted" Codd (1923–2003)

"A Relational Model of Data for Large Shared Data Banks"
Communications of the ACM 13 (6): 377–387, June 1970

Group exercise #1

Design the schema for a "zoo" database

Relations

- animal
- building
- cage
- keeper
- species

Consider

- What attributes needed for each relation?
- How will you join the relations together?

Primary/Foreign Key Constraints

Keys of relations

Unique key

- Attribute (or set of attributes) such that
- No pair of tuples have the same value(s)

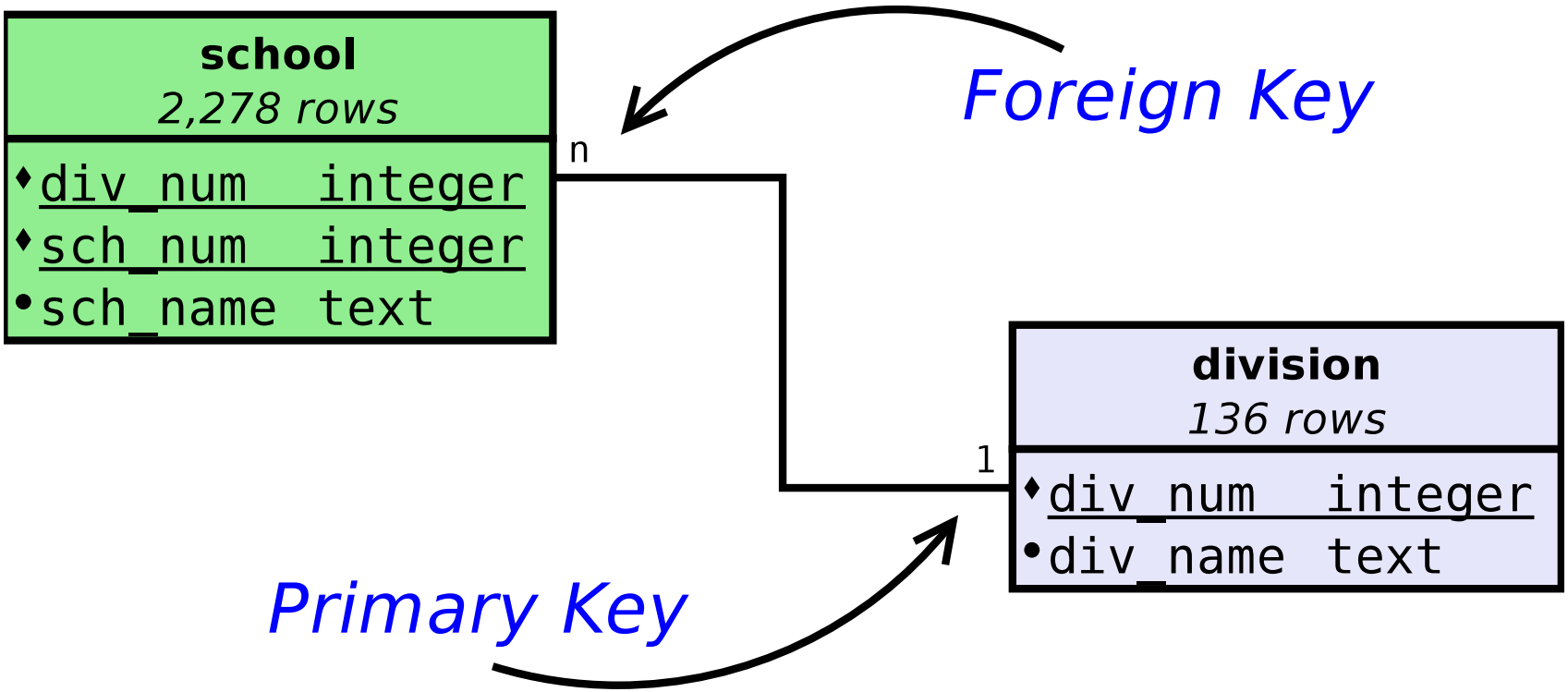
Useful for identifying / referencing tuples

What are the keys in:

Student	Course	Grade
Hermione Grainger	Potions	A-
Draco Malfoy	Potions	B
Harry Potter	Potions	A
Ron Weasley	Potions	C

Key attributes are often underlined or **bolded**

Example from vdoe



PRIMARY KEY syntax

Single attribute:

```
CREATE TABLE MovieStar (
    name char(30) PRIMARY KEY,
    address varchar(255),
    gender char(1) NOT NULL,
    birthdate date
);
```

Multiple attributes:

```
CREATE TABLE Movies (
    title char(100),
    year integer,
    length integer NOT NULL,
    genre char(10) NOT NULL,
    studioName char(30),
    producerC# integer,
    PRIMARY KEY (title, year)  -- tuple syntax
);
```

Foreign keys

Referenced attributes must be a `PRIMARY KEY` or `UNIQUE`

```
CREATE TABLE Studio (
    name char(30) PRIMARY KEY,
    address varchar(255),
    presC# integer REFERENCES MovieExec(cert#)
);
```

Alternatively / for multiple attributes:

```
CREATE TABLE Studio (
    name char(30) PRIMARY KEY,
    address varchar(255),
    presC# integer,
    FOREIGN KEY (presC#) REFERENCES MovieExec(cert#)
);
```

Group exercise #2

Declare keys for your "zoo" database

Relations

- animal
- building
- cage
- keeper
- species

Consider

- What are the PRIMARY KEYS?
- What are the FOREIGN KEYS?

Attribute and Tuple Constraints

Attribute constraints

May be declared on one or more attributes

`NOT NULL`

- These attributes never have `NULL` values
- Use this constraint whenever possible!

`UNIQUE`

- No two tuples may have the same values (except `NULL`)

`PRIMARY KEY` = `UNIQUE` + `NOT NULL`

- Each relation may only have one
- Every relation should have one!

Attribute-based CHECK constraints

New requirement: all certs must be 6+ digits

```
CREATE TABLE Studio (
    name char(30) PRIMARY KEY,
    address varchar(255),
    presC# integer CHECK (presC# >= 100000)
);
```

- Only checked when presC# is inserted/updated

What's the difference between a *data type* and a *domain*?

```
CREATE TABLE MovieStar (
    name char(30) PRIMARY KEY,
    address varchar(255),
    gender char(1) CHECK (gender IN ('F', 'M', 'O'))
);
```

- Only checked when gender is inserted/updated

CHECK ≠ FOREIGN KEY

What's wrong with this relation?

```
CREATE TABLE Studio (
    name char(30) PRIMARY KEY,
    address varchar(255),
    presC# integer CHECK
        (presC# IN (SELECT cert# FROM MovieExec))
);
```

- Only checked when `presC#` is inserted/updated
- `NULL` values will also cause this `CHECK` to fail

Is this a bug or a feature?

Group exercise #3

Constraints for your "zoo" database

Relations

- animal
- building
- cage
- keeper
- species

Consider

- Attribute CHECK constraints?
- Tuple-based CHECK constraints?